IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Appellants:	Bala, et al.	Patent Application	
Application No.:	09/874,170	Group Art Unit:	2123
Filed:	June 4, 2001	Examiner:	Proctor, J.

For: NETWORKED CLIENT-SERVER ARCHITECTURE FOR TRANSPARENTLY TRANSFORMING AND EXECUTING APPLICATIONS

REPLY BRIEF

In response to the Examiner's Answer mailed on March 31, 2008, Appellants respectfully submit the following remarks.

REMARKS

Appellants are submitting the following remarks in response to the Examiner's Answer. In these remarks, Appellants are addressing certain arguments presented in the Examiner's Answer. While only certain arguments are addressed in this Reply Brief, this should not be construed that Appellants agree with the other arguments presented in the Examiner's Answer.

Response to Argument on Pages 9, First Paragraph, through Page 14, Second Paragraph of the Examiner's Answer

In the Appeal Brief filed February 11, 2008, while addressing a 35 U.S.C. §103(a) rejection of Claims 2-24, and in particular independent Claim 2, Appellants argue that Shimura (U.S. Patent Application No. 6,370,687)(hereinafter, Shimura) does not:

> teach dynamic parsing of application code into code segments wherein the parsing of the code segments is dynamically performed based on actual server-side and client-side execution overhead, network bandwidth efficiency and client-side storage requirements on a per client basis

(emphasis in original; Appeal Brief, page 10, last paragraph – page 11, first paragraph).

In the response to this argument, the Examiner's Answer asserts that "Shimura discloses dynamic parsing of application code" (emphasis in original; Examiner's Answer, page 9, last paragraph). Additionally, the Examiner's Answer states, "'Parsing' is a necessary and inherent step to compile application code such as Java code and there appears to be no argument that generic 'parsing' is disclosed in Shimura" (Emphasis added; Examiner's Answer, page 10, first paragraph). Furthermore, the Examiner's Answer provides:

> Thus Shimura discloses dynamic parsing of application code within the meaning of the claim language. Further, Appellants' specification fails disclose a deliberate or special definition of 'dynamic parsing' beyond what the Examiner has shown above. If, to Appellants, 'dynamic parsing' means something other than parsing in real time and in response to a real time request, that definition is not found in the specification.

(Emphasis in original; Examiner's Answer, page 10, third paragraph.)

"As reiterated by the Supreme Court in *KSR*, the framework for the objective analysis for determining obviousness under 35 U.S.C. 103 is stated in *Graham v. John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966). Obviousness is a question of law based on underlying factual inquiries" including "[a]scertaining the differences between the claimed invention and the prior art" (MPEP 2141[II]). "In determining the differences between the prior art and the claims, the question under 35 U.S.C. 103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious" (emphasis in original; MPEP 2141.02[I]). Appellants note that "[t]he prior art reference (or references when combined) need not teach or suggest all the claim limitations, however, Office personnel must explain why the difference(s) between the prior art and the claimed invention would have been obvious to one of ordinary skill in the art" (emphasis added; MPEP 2141[III]).

Additionally, MPEP §2141.02 VI provides, "[a] prior art reference must be considered in its entirety, i.e., as a whole, including portions that would lead away from the claimed invention" (emphasis added; MPEP 2141.02 VI, *W.L. Gore & Associates, Inc. v. Garlock, Inc.*, 721 F.2d 1540, 220 USPQ 303 [Fed. Cir. 1983], *cert. denied*, 469 U.S. 851 [1984]).

Appellants respectfully submit that Appellants' claimed invention as a whole would not have been obvious, and therefore the Examiner's Answer does not satisfy the requirements for a rejection of Claims 2-24 under 35 U.S.C. §103(a). In particular, Appellants respectfully submit that the Examiner's Answer fails to explain the differences between Shimura and Appellants' claimed features, in which portions of Shimura lead away from Appellants' claimed features. Moreover, Appellants respectfully submit that the Examiner's Answer fails to explain why these differences would have been obvious to one of ordinary skill in the art.

Appellants respectfully submit that the teachings of Shimura lead away from Appellants' claimed features. In particular, Appellants understand Shimura to teach "...a compiling function is imparted to a substitute server so that a computer system and a substitute compile server are provided which allows a client's high-speed execution of a virtual machine computer program lying on a network" (Shimura, Column 2, lines 10-15). Additionally, in one example of Shimura's operation, Shimura states in Column 6, lines 25-30 the following:

> Then, in response to a request for the Java program from a client to the Web server on the network, the substitute compile server 10 returns to the client the Java program which has been compiled and optimized into a native code conforming to the execute for of the requester client.

(Emphasis added.) Similarly, Column 4, lines 42-51 of Shimura states:

> When the Java program is delivered from the Web server 20 to the substitute compile server 10, the compile controller 22 provides a control of the compile unit 28 so as to allow the byte code of the Java program delivered from the Web server 20 to be compiled into a machine code of native environment conforming to the execution environment of the client issuing the request at that time. Simultaneously, the compile controller 22

> retains the thus compiled Java program in the cache unit 12 and enters the cache results into the cache table 24.

(Emphasis added.)

Of particular importance to note, Shimura only describes its compilation of the requested program to be the compilation of the whole program, to be cached as a whole. There is no mention of "parsing the client application in the native binary format into a plurality of code segments" as is provided in the features of Appellants' Claim 2. In fact, Shimura remains silent as to parsing or dividing a client application into a plurality of code segments. Shimura's objective through its compilation of an entire program is to enable a later execution to become "feasible at a high speed" (Shimura, Column 2, lines 54-57).

As stated above, the Examiner's Answer asserts that "parsing" is a necessary and inherent step to compile application code". Additionally, the Examiner's Answer asserts that Appellants have presented no argument that Shimura is merely doing generic "parsing".

Appellants respectfully submit that the term 'parsing' within Claim 2 must be read in context with the rest of the terms within Claim 2. For example, Claim 2 states, "for parsing the client application in the native binary format into a plurality of code segments", and "a code cache coupled to the CPU, for storing said code segments", and the following:

> …a client code manager coupled to the code cache for launching the client application by requesting that the server code manager transmit at least one of the plurality of dynamically tailored code segments to the client, receiving at least one of the

dynamically tailored code segment form the server, storing the dynamically tailored code segments in the code cache,…

As shown, the features of Claim 2 do not merely 'parse' application code. The features of Claim 2 parse a client application into a plurality of code segments, and then store at least one of the plurality of code segments in a code cache.

Additionally, Appellants' specification describes different scenarios in which one would want to parse a client application into a plurality of code segments and then store at least one of the plurality of code segments in a code cache. (See Appellants' specification, pages 9-10.) Some examples include parsing a client application into a plurality of code segments and storing at least one of the plurality of code segments in a code cache for security and for convenience when an online connection is not stable.

While Shimura teaches compiling and storing a program as a whole for the purpose of providing "high-speed execution of a virtual machine computer program lying on a network", Appellants' Claim 2 provides for "parsing the client application in the native binary format into a plurality of code segments", and "storing the dynamically tailored code segment in the code cache". Appellants respectfully submit that Shimura teaches away from the features of Appellants' Claim 2.

As presented above, Appellants respectfully submit that Shimura fails to suggest "parsing the client application in the native binary format into a plurality of code segments" and "storing the dynamically tailored code segment in the code cache" as is recited in Appellants' Claim 1.

In fact, Appellants respectfully submit that Shimura teaches away from Appellants' claimed features. Additionally, Appellants respectfully submit that the Examiner's Answer fails to explain why the differences between Shimura and Appellants' claimed features would have been obvious to one of ordinary skill in the art.

CONCLUSION

In view of the above remarks, Appellants continue to assert that Shimura does not suggest Appellants' claimed features, for reasons presented above and for reasons previously presented in the Appeal Brief.

Respectfully submitted,

WAGNER BLECHER

Dated: 06/02/2008

/John P. Wagner, Jr.
John P. Wagner, Jr.
Registration Number: 35,398

WAGNER BLECHER
123 Westridge Drive
Watsonville, CA 95076
(408) 377-0500